**Tcpdump Lab**

**Introduction:**

**tcpdump** - is a command-line utility used primarily for packet analysis. It captures packet data from network interfaces and can use the Berkley Packet Filter (BPF) syntax. The key point to remember here is that it is a lightweight packet analysis tool and it is a common suite used by many organizations to conduct packet captures (PCAPs).

**Some of the most common parameters used with tcpdump:**

-#: Display line/packet number  
-c count: Number of packets to capture before tcpdump automatically exits.  
-D: Display interfaces  
-e: Display Ethernet header data.  
expression: Specify a Berkeley Packet Filter (BPF) statement to filter traffic.  
-i interface: Specify from which network interface you would like tcpdump to sniff. This generally requires administrative permissions.  
-n: Don't resolve hostnames or well-known port numbers to their service.  
-r file: Specify an existing pcap file to read from instead of a network interface.  
-s snaplen: Snapshot length, or the number of bytes to capture per packet. Default is 262,144 bytes but this may vary across platforms.  
-w file: Specify a new pcap file to place filtered packets in.  
-X: Show packet contents in hexadecimal and ASCII.  
-v : Display verbose output

**Task 1: Examining Packet Headers**

1.1) We will start out by using tcpdump to examine packet headers. For example, if we wanted to display headers and line numbers of the first 20 packets, we would type the following command

***Command:*** *tcpdump -n -r investigate.pcap -c 20 -#*

*A screenshot of a computer

Description automatically generated*

**Notes:** (Part of the image is cut off, but for the sake of this example, we will focus on the command input that gave us the output above.)

1. The “tcpdump” command above tells the system that we want to use tcpdump
2. The “-n” command tells the system that we do NOT want hostnames or well-known port numbers resolved to their services.
3. “-r investigate.pcap” tells the system we want to open this file. When you type “-r” remember to specify the file you want to open immediately after.
4. -c 20” Tell the system that we want to specify the number of packets to display. In this example, we chose to display 20 packets.
5. “-#” is used to number the packets neatly and clearly so that they are easier to read.

1.2) In the output of the previous step, each line represents a single packet and shows the details associated with it. The following chart will help you understand how to read the output of these packets.

|  |  |
| --- | --- |
| **Field Output** | **Description** |
| 1-10 | Line number/Packet number |
| 21:23:57.196268 | Timestamp |
| IP | (Layer 3) Protocol being used |
| 10.130.8.94 | Source IP address |
| 57810 | Source port |
| > | Data flow indicator |
| 10.130.8.2 | Destination IP address |
| 53 | Destination port |
| 44934+ [1au] PTR? 94.8.130.10.in-addr.arpa. | DNS information |

Reference:

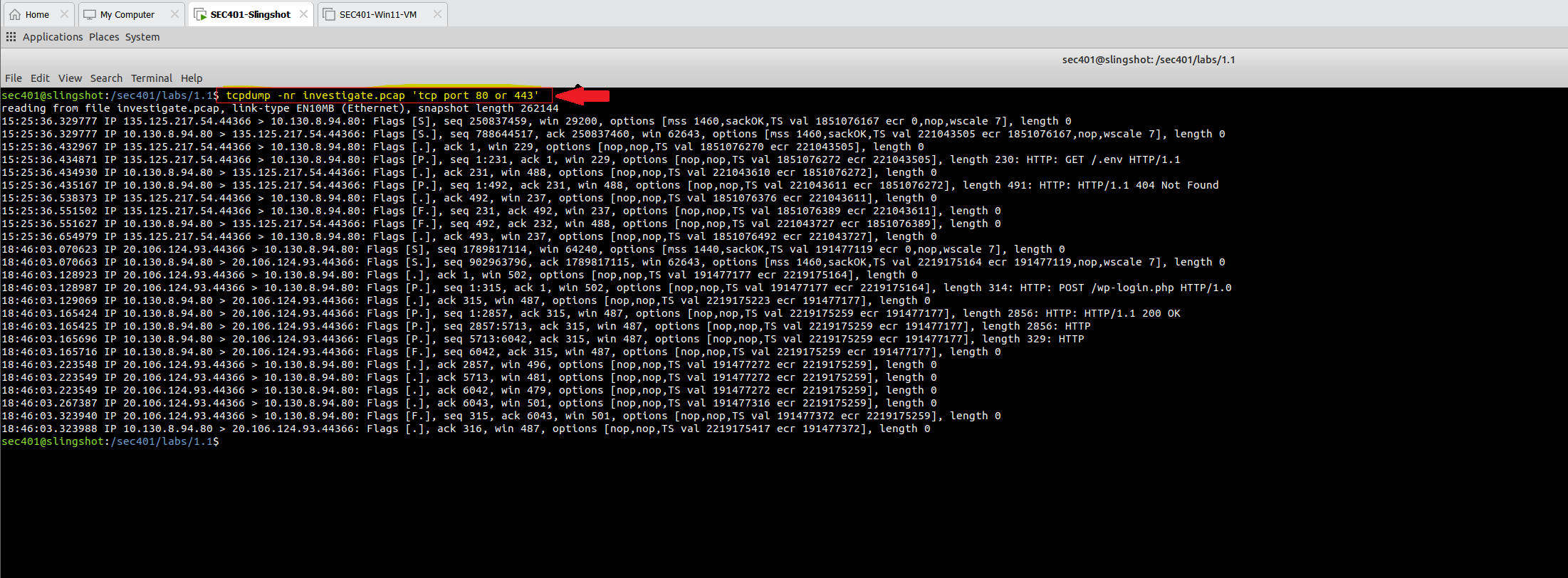
* Tcpdump. “man page” <https://www.tcpdump.org/manpages/tcpdump.1.html>

**Task 2: Filtering Traffic from Packet Captures (pcaps)**

2.1) To filter packets and achieve more in-depth granular control, we will use the Berkeley Packet Filter (BPF) method to identify specific hosts and/or protocols to which traffic is going to or coming from. The BPF is essentially a form of a network “tap” that filters traffic/packets based on its parameters between layers 2-4. It can work in conjunction with tcpdump to provide extreme filtration and precision. It is a great tool for intrusion detection analysis.

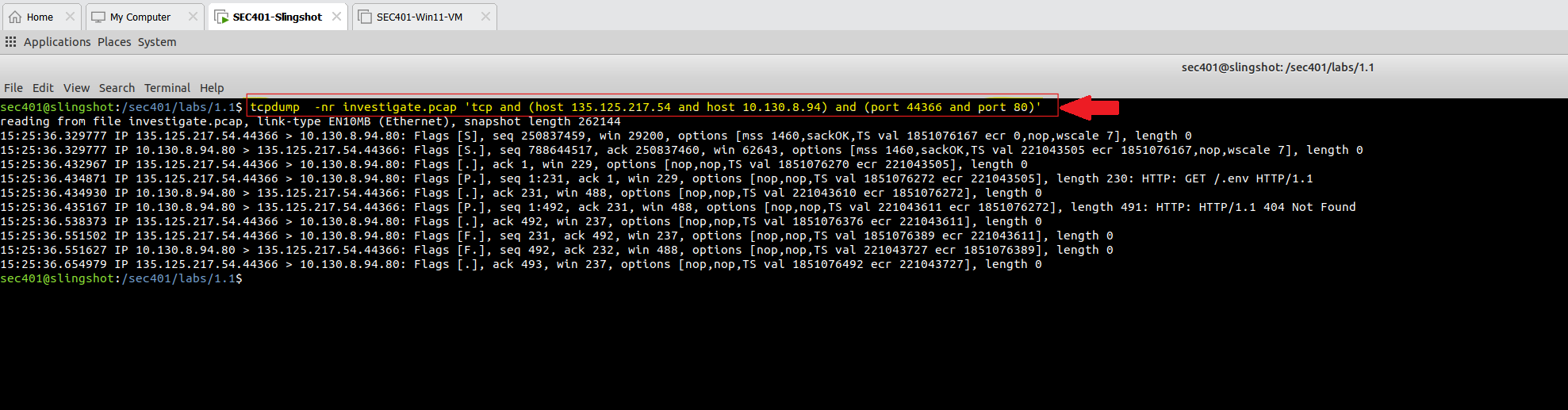
a.) Let's say that we want TCP traffic from ports 80 or 443. We would type the following command

***Command:*** *tcpdump -n -r investigate.pcap ‘tcp port 80 or 443’*



b.) Now, let's say that we want TCP traffic from ports 44366 and 80 that is between hosts 135.125.217.54 and 10.130.8.94. We would type the following command

***Command:*** *tcpdump -n -r investigate.pcap ‘tcp and (host 135.125.217.54 and host 10.130.8.94) and (port 44366 and port 80)’*



2.2) In the above examples, we used tcpdump in conjunction with the BPF syntax to specify the information we want.

In the first image, we specified a packet capture file and then went on to specify that we ONLY wanted tcp traffic on ports 80 (HTTP) and 443 (HTTPS).

In second image, we dug a little deeper. Not only did we specify that we wanted tcp traffic, but this time we specified on what hosts we wanted that traffic coming from and going to AND we also specified what ports we wanted it on.

As you can see, we can get very particular with these tools. This is a way to help you drill down in your detection and analysis to find the exact information you’re looking for.

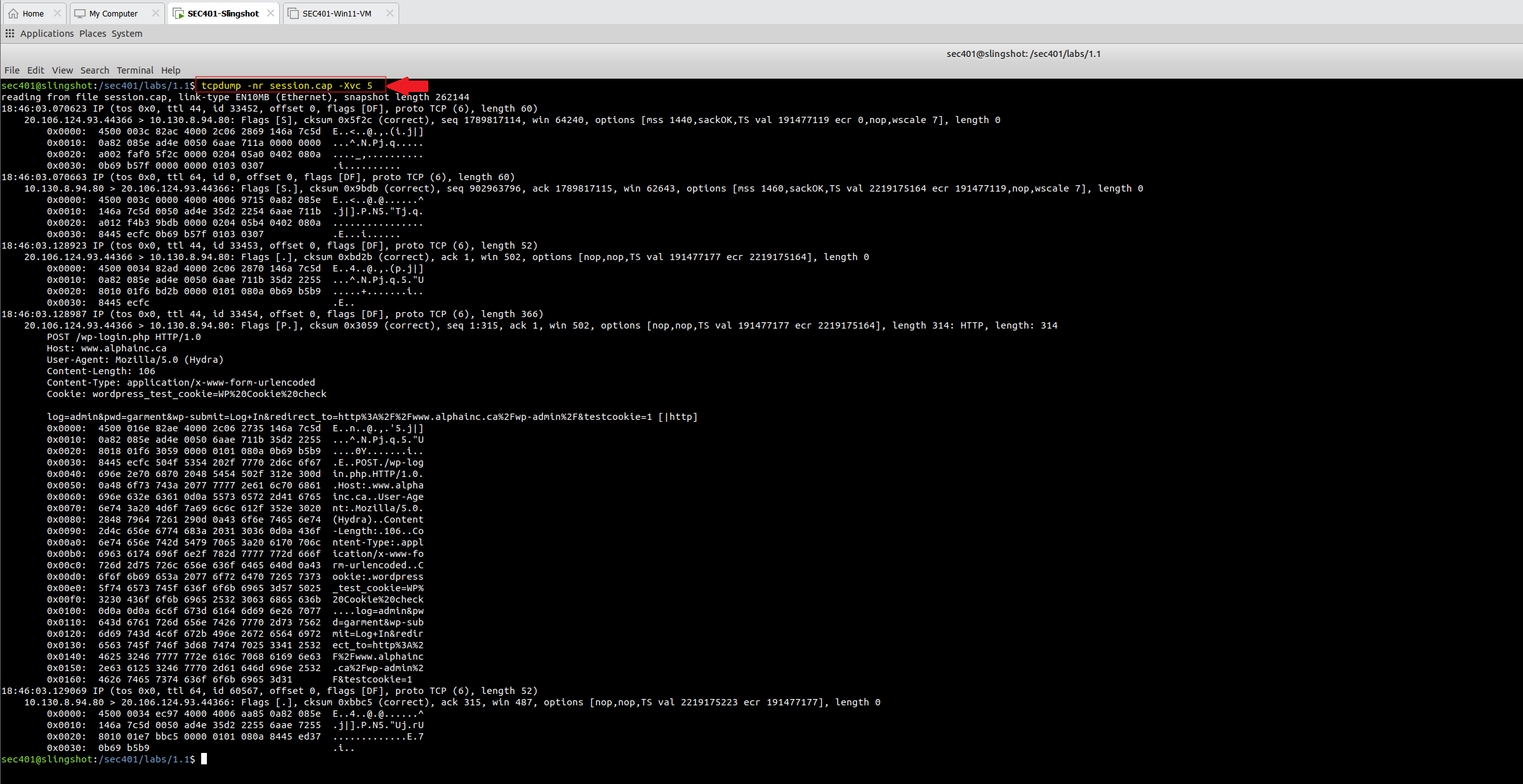
References:

* https://www.ibm.com/docs/en/qsip/7.4?topic=queries-berkeley-packet-filters
* https://docs.securityonion.net/en/2.4/bpf.html

**Task 3: Examine Raw Packet Content**

3.1) tcpdump is also able to display raw packet information in various formats (hexdump etc). Let’s try examining the first 5 packets in hexdump format from another pcap file.

***Command:*** *tcpdump -nr session.pcap -Xvc 5*

****

If you’ve noticed, I've been shortening/chaining together commands to save time. For example, instead of typing (*tcpdump -n -r session.pcap -X -v -c 5),* we can just combine the majority of the commands like so (*tcpdump -nr session.pcap -Xvc 5)* and get the same results. As long as the commands make sense to the operating system, you will be good to go!

That is the power and flexibility of Linux! You can become rather creative the more you play around with it. Find commands, or shortcuts, that work for you and help enhance your efficiency!

**Conclusion:**

That brings our brief tutorial on tcpdump to a close. Please feel free to give any comments or feedback if this lesson provided you with any value, and stay tuned for more labs in the future. Thank you!